

Papyrus for RealTime

- Out of its shell

Charles Rivet

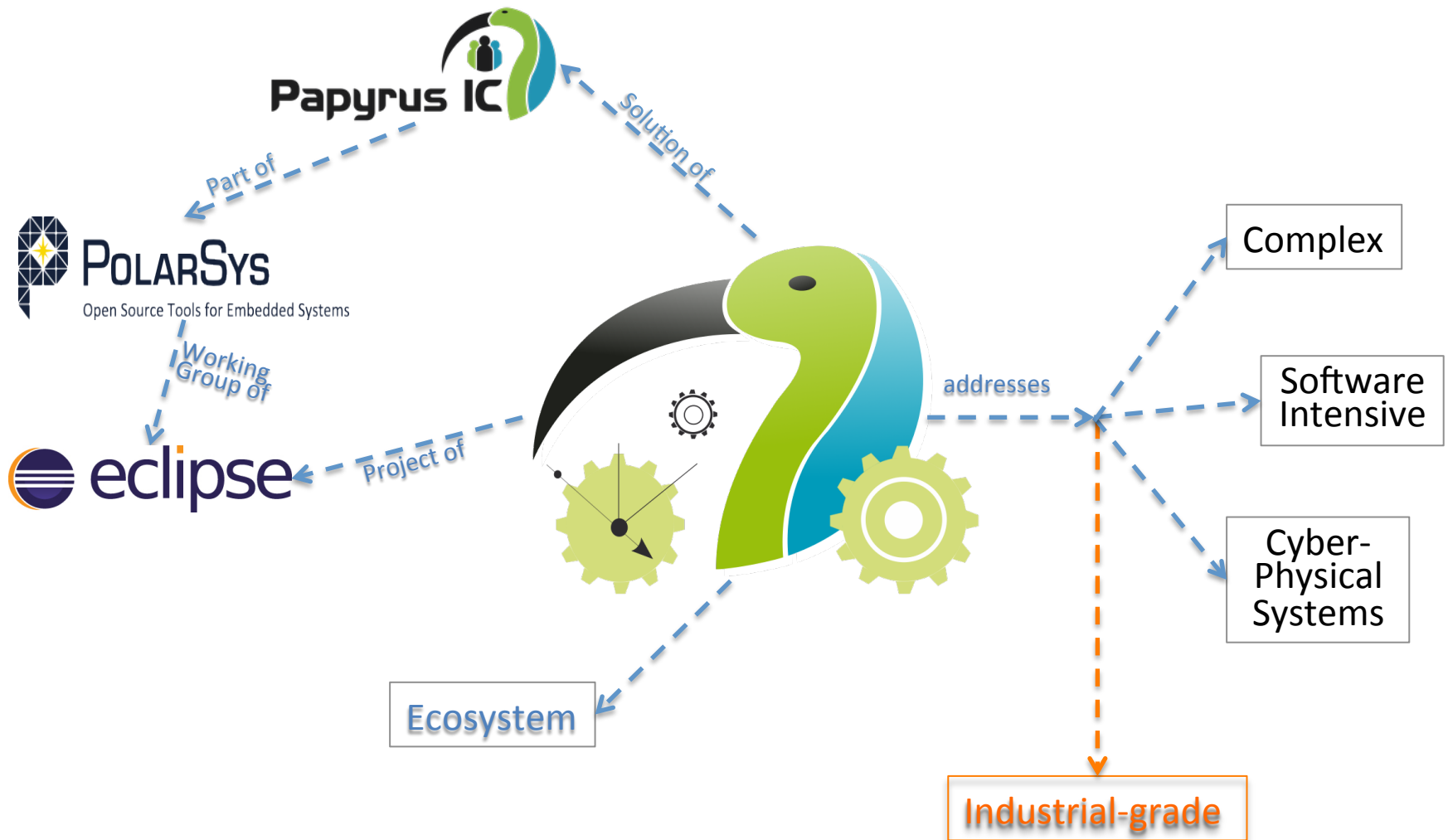
Senior Product Manager, Papyrus-RT product lead

Zeligsoft

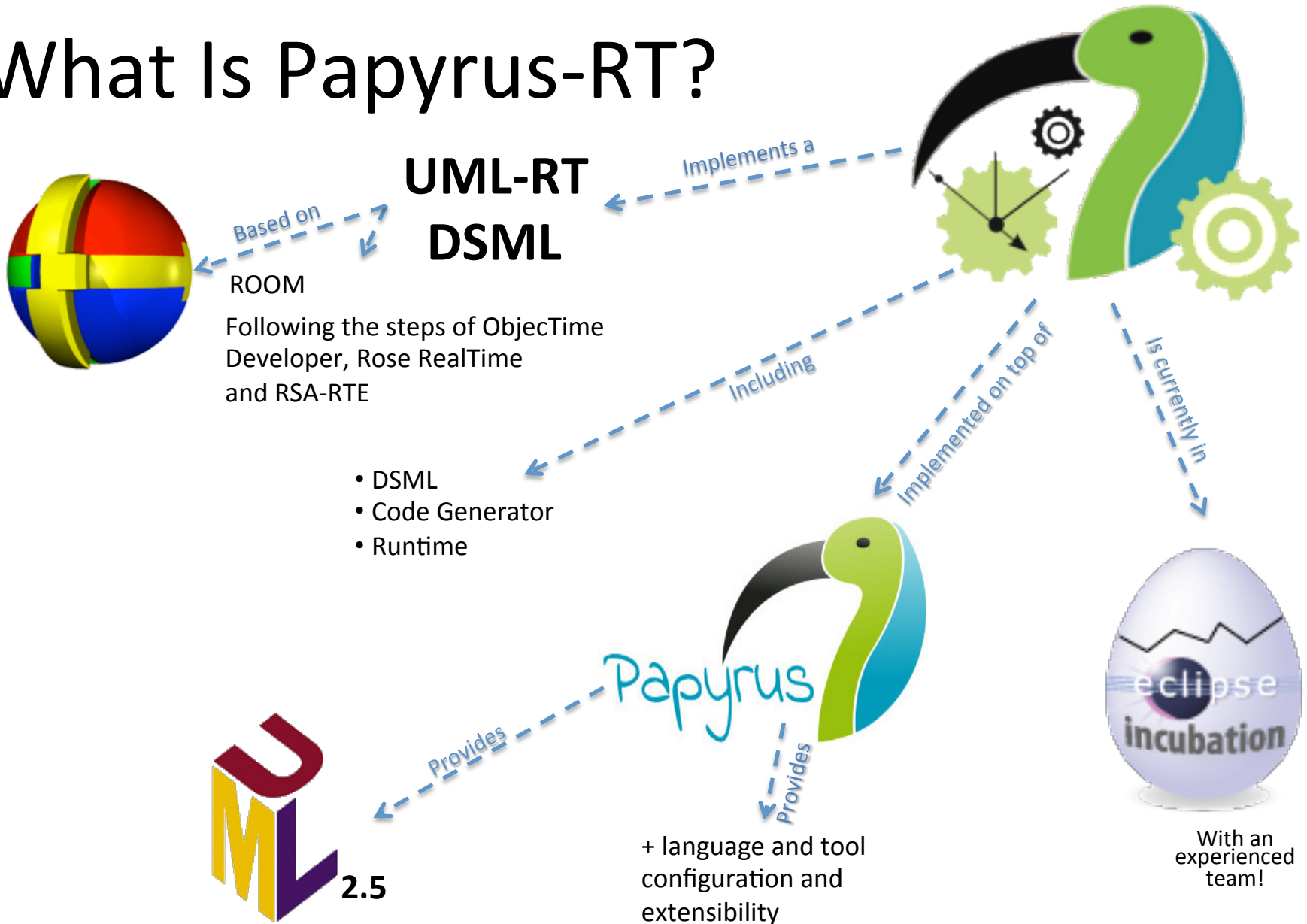
charles@zeligsoft.com



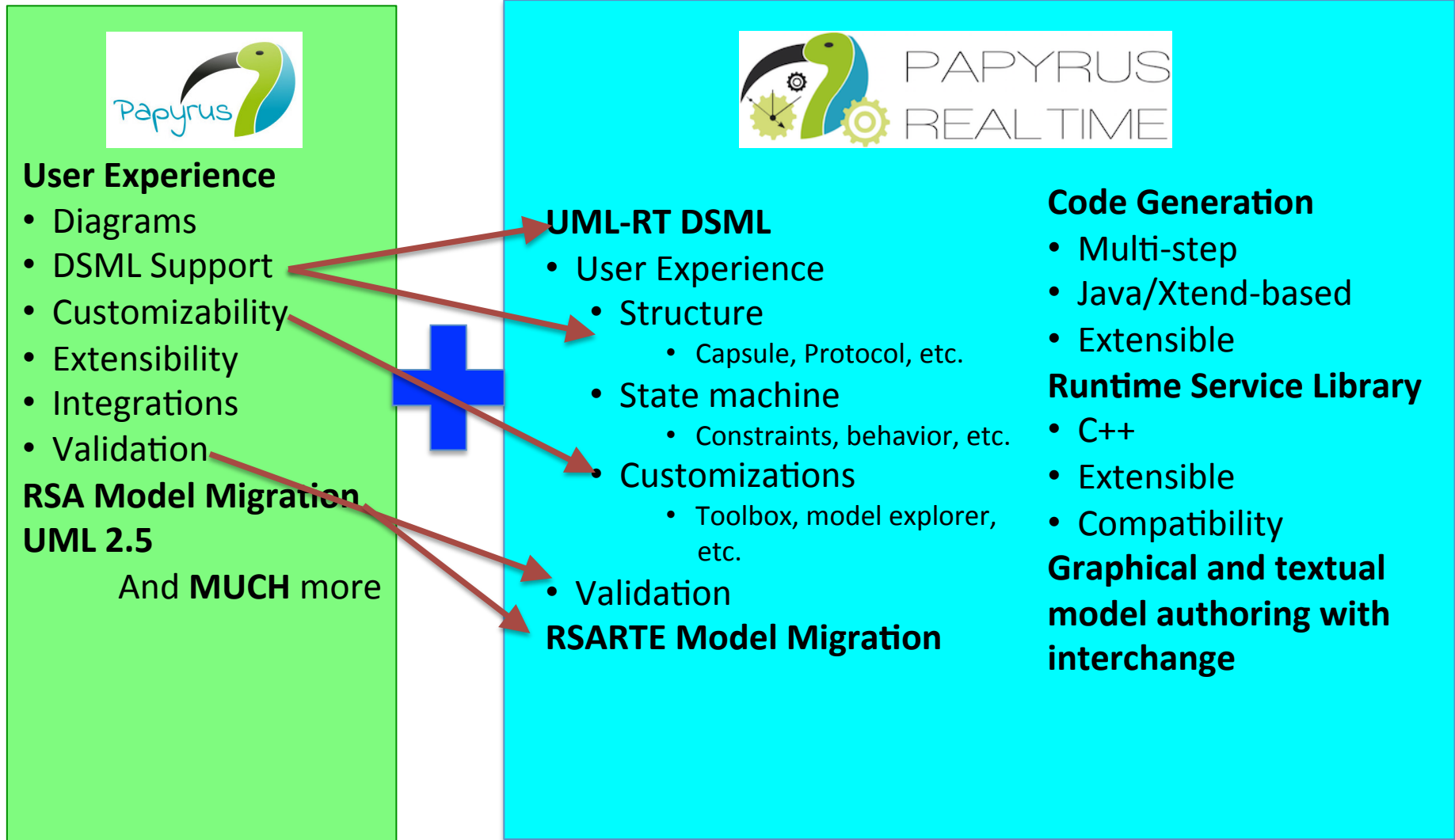
Where does Papyrus-RT fit?



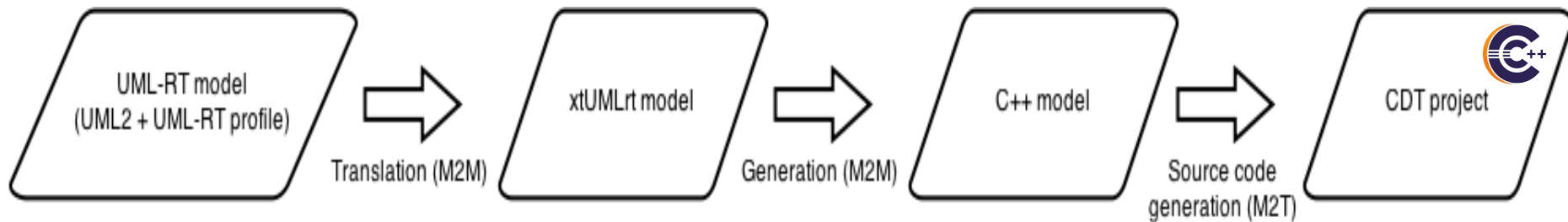
What Is Papyrus-RT?



Architecture - Overview

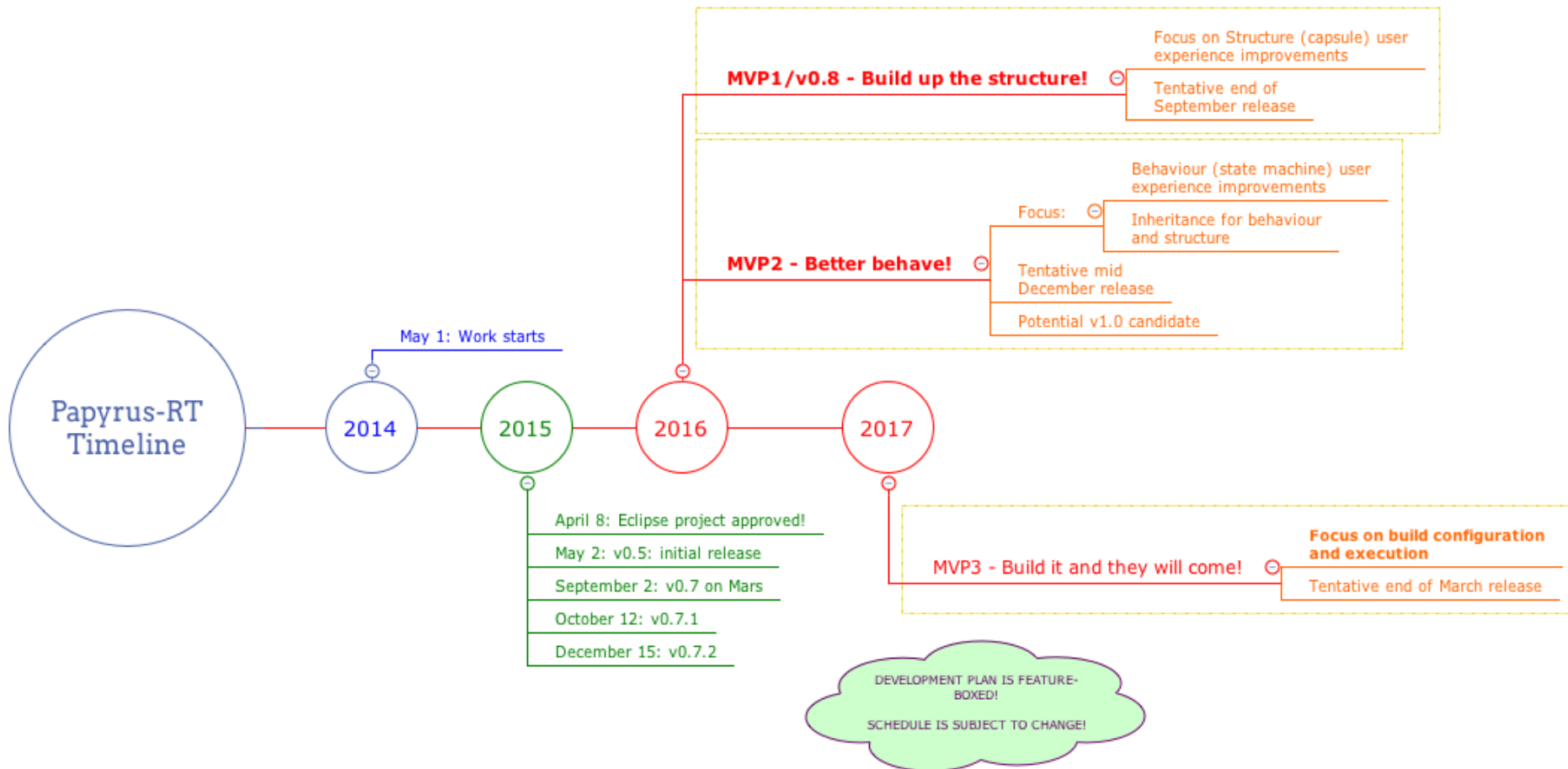


Architecture – Code Generation



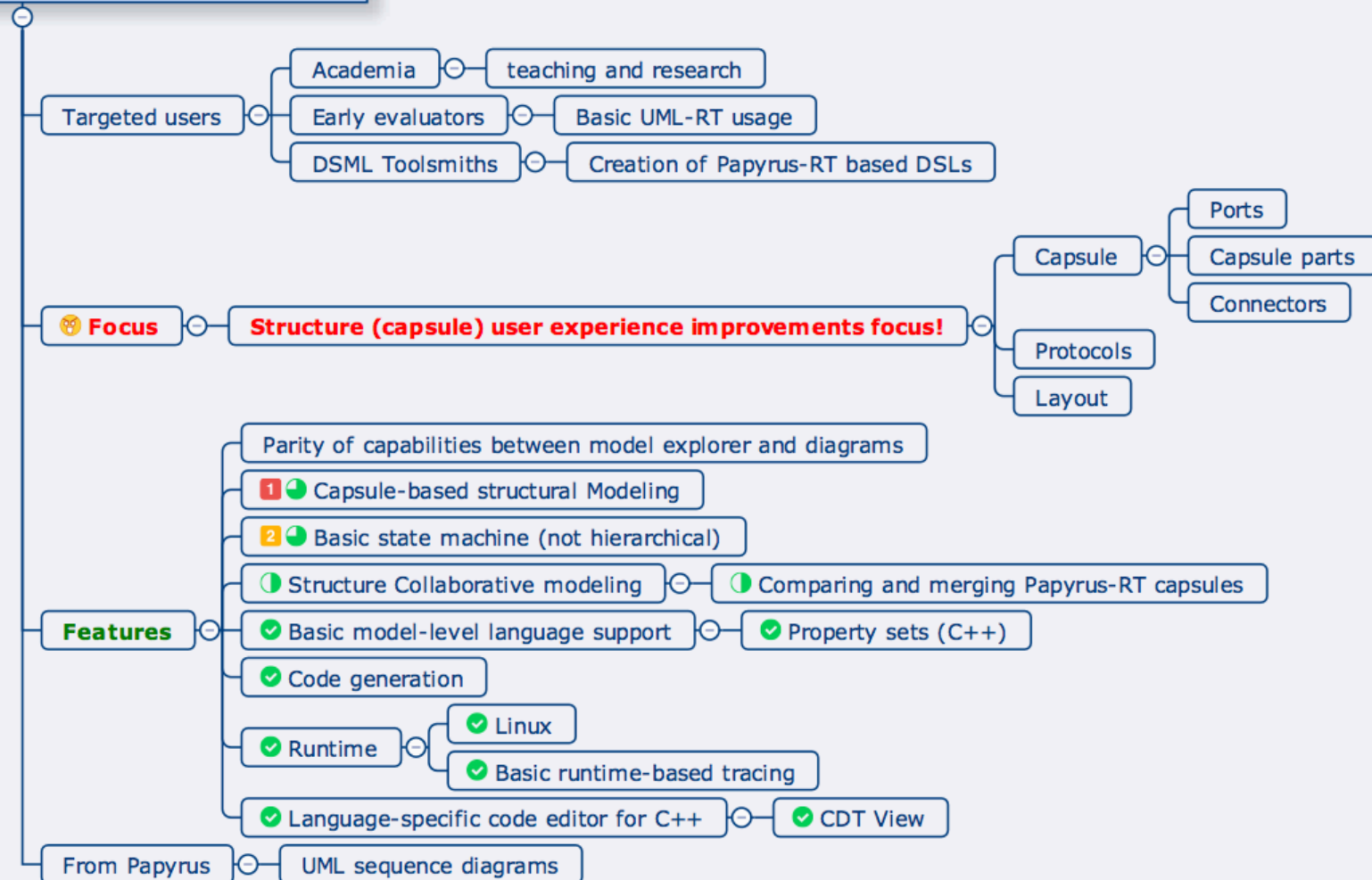
- Implemented in Java and **Xtend**
- Supports incremental generation
- Targeting C++03 on Linux, Windows, and Mac

Papyrus-RT Timeline



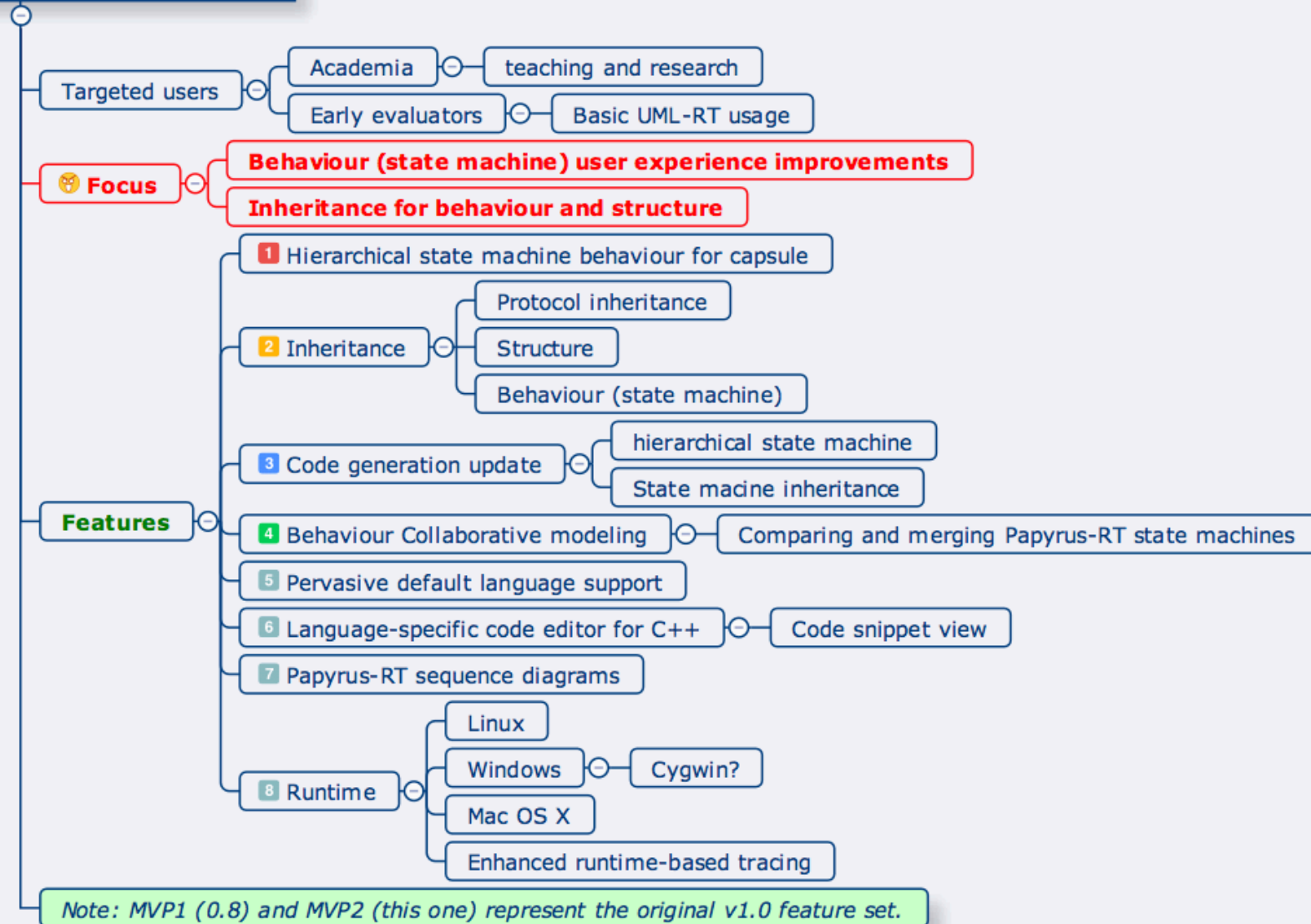
Papyrus-RT MVP1

1 MVP1 - Build up the structure! (v0.8)



2 MVP2 - Better behave! (v0.9 - v1.0 candidate)

Papyrus-RT MVP2



Demo ? Workshop !

This room!

@13:45!

90 minutes of fun!

Thank you!



Who is Papyrus-RT?



Tooling



Tooling, validation, import, CDT integration



Collaborative Modeling



Requirements, C++ profile, intermediate representation, testing



Intermediate representation



UML-RT Profile



Code generation, runtime, CDT integration

And more to come...

Papyrus-RT Links of Interest

Home	https://eclipse.org/papyrus-rt/
Project	https://projects.eclipse.org/projects/modeling.papyrus-rt
Wiki	https://wiki.eclipse.org/Papyrus-RT (*)
Releases	https://eclipse.org/papyrus-rt/content/download.php
Forum	http://bit.ly/PapyrusRTCommunity
Blog	https://papyrusuml.wordpress.com/
Twitter	https://twitter.com/papyrusuml
Consortium	https://www.polarsys.org/ic/papyrus



Runtime – Directory Layout

```
+ rts
+ build
  + buildtools          // Toolchain-specific makefile fragments
    + x86-gcc-4.6.3
    + x86-gcc-4.6.3-debug
    + x86-VisualC++-12.0
  + os                  // OS-specific makefile fragments
    + linux
    + windows
+ include               // External-facing (API) include files
+ Makefile              // Makefile to build the runtime
+ obj                   // Object and library files for the runtime
  + linux.x86-gcc-4.6.3
    + os                // OS-specific object
    + umlrt             // Runtime objects
    + util              // Runtime utilities objects
+ os                    // Source files for OS-specific functions (e.g., thread, mutex, time, etc.)
  + linux               // Source files for linux-specific implementations
  + windows             // Source files for Windows-specific implementations
+ umlrt                 // UML-RT Runtime sources (.cc)
+ util                  // Runtime internal utilities (low-level logging for debug)
+ tests                 // Runtime tests
```

Hybrid textual-graphical Modeling

Graphical Representation

- Clearly show the relationship between elements
- Improves comprehension and understanding in some cases, such as with state machines or with the structure of an application

Textual Representation

- Faster model creation
- Can be modified with standard light weight editors or with a language aware editor
- Standard CM tools and diff/merge tools can be used. For graphical, the diff/merge must be language aware to provide reasonable performance

Why not have both and allow the user to choose based on what is best for their needs at the time or task involved?